Mathematical Biosciences and Engineering

MBE, 20 (2): 4178–4197.
DOI: 10.3934/mbe.2023195
Received: 18 September 2022
Revised: 29 November 2022
Accepted: 29 November 2022
Published: 21 December 2022

http://www.aimspress.com/journal/MBE

Research article

Generating new protein sequences by using dense network and attention mechanism

Feng Wang[1,2], Xiaochen Feng[2], Ren Kong[3] and Shan Chang[3,*]

[1] School of Computer Engineering, Suzhou Vocational University, Suzhou, China
[2] Information Engineering Department, Changzhou University Huaide College, Taizhou, China
[3] Institute of Bioinformatics and Medical Engineering, Jiangsu University of Technology, Changzhou, China

* **Correspondence:** Email: schang@jsut.edu.cn.

Abstract: Protein engineering uses de novo protein design technology to change the protein gene sequence, and then improve the physical and chemical properties of proteins. These newly generated proteins will meet the needs of research better in properties and functions. The Dense-AutoGAN model is based on GAN, which is combined with an Attention mechanism to generate protein sequences. In this GAN architecture, the Attention mechanism and Encoder-decoder can improve the similarity of generated sequences and obtain variations in a smaller range on the original basis. Meanwhile, a new convolutional neural network is constructed by using the Dense. The dense network transmits in multiple layers over the generator network of the GAN architecture, which expands the training space and improves the effectiveness of sequence generation. Finally, the complex protein sequences are generated on the mapping of protein functions. Through comparisons of other models, the generated sequences of Dense-AutoGAN verify the model performance. The new generated proteins are highly accurate and effective in chemical and physical properties.

Keywords: generative adversarial network; encoder-decoder; dense network; protein sequence

1. Introduction

With the development of protein engineering [1–4], the purposeful design and creation of new proteins [5–8] have become an important research topic in the field of bioinformatics. The

establishment of proteins with required function is one of the essential tasks for molecular biology [9–12]. A protein is composed of an amino acid sequence [13,14]. The structure and function of the protein are derived from the specific geometric arrangement of its linear amino acid sequence. Therefore, the amino acid sequence will affect the structure, function and physicochemical properties of the protein. Solving the problem of protein sequence will help to solve the functional problem of proteins [15,16]. De novo protein generation techniques to generate rational protein sequences with natural physicochemical properties help provide breakthrough strategies for biomedical and life sciences.

With the development of de novo protein design, traditional algorithms have been widely used in sequence optimization and protein prediction. In the early stage, Lee et al. [17] proposed a Ga-ACO algorithm, which is a new algorithm with ACO [18] Genetic algorithm (GA) [19]. The GA is improved by combining local search [20] and ant colony optimization for multiple sequence alignment. The GA-ACO algorithm performs a genetic algorithm to provide a variety of comparisons, and ant colony optimization is performed to remove local optima. Issa [21] proposed the Asca-pso algorithm, and a sine cosine algorithm (SCA) [22] was integrated with particle swarm optimization (PSO) [23] for matching paired local sequences. PSO makes better use of the search space than standard SCA operators, which are better suited to finding the longest continuous substring between two biological sequences. Asca-pso solutions provide excellent performance in terms of accuracy and computation time. Zhan et al. [24] proposed a multi-sequence alignment algorithm named ProbPFP that combines a partition function [25] and hidden Markov model [26] optimized by particle swarm integration [27]. The alignment accuracy was further improved. In addition, SVM [28] is often used in protein prediction: for example, protein structure prediction based on bilayer SVM [29]. These algorithms laid the foundation for the early development of protein engineering. Chou et al. [30] proposed a new real space method, based on the principles of simulated annealing, for determining protein structures on the basis of interproton distance restraints derived from NMR data. Yanover et al. [31] proposed a variant of the standard dead-end elimination (DEE), called type-dependent DEE. The method reduces the size of the conformational space of the multistate design problem, while provably preserving the minimal energy conformational assignment for any choice of amino acid sequence.

In recent years, deep learning [32,33], as a new machine learning technology, has also gained attention from bioinformatics due to its strong learning ability. It is used to predict protein structure [34] and solve protein sequence problems. Classical deep learning methods usually use the architecture of multi-layer convolutional neural network (CNN) [35] to train models. For example, one-dimensional convolutional neural network [36] is used to study folding recognition and sequence-structure relations. The protein secondary structure is predicted by using deep convolutional networks [37]. Wang et al. [38] effectively combined deep learning with traditional protein function prediction algorithms and predicted protein ordered/disordered regions by weighted deep convolutional neural fields. Generative adversarial network (GAN) [39] is a deep learning technology and an unsupervised deep learning model. It consists of two adversarial networks, generator network and discriminator network. Due to its unique learning mode, it is often used for image generation and gene expression. Generators generate fake samples and try to trick the discriminator network into believing that the generated samples are real. The discriminator tries to distinguish the generated samples from the real ones and direct the generator network to produce more realistic samples. In the framework of generative adversarial networks (GAN), many aspects of drug design and discovery have been studied [40]. The original protein sequences are used as data sets to generate new protein sequences. In terms of protein feature extraction, Wan et al. [41] proposed a new method, ffpred-gan, based on generative adversarial

networks. It can accurately learn the high-dimensional distribution of biophysical features through protein sequences and generate high-quality synthetic protein feature samples. Gupta et al. [42] proposed feedback GAN (FBGAN). It is a kind of architecture that utilizes a feedback mechanism to optimize protein function and synthesize proteins with variable length encoding of DNA sequences.

In this work, we propose a deep learning architecture model based on GAN. Our proposed model, called Dense-AutoGAN, is used to generate protein sequences. Dense-AutoGAN's generator network feeds sequence data into an "encoder-decoder" via an Attention mechanism [43] to accurately capture the structural representation and potential characteristics of protein sequences. At the same time, another part of the discriminator of the model is also learning the difference between the generated protein sequence and the real sequence, so that the sequences generated by the generator network are closer to the real sequence. In addition, a method of generator stacking for the GAN internal network structure is proposed to expand the training space of the discriminator and further improve the efficiency of generating sequence. In this study, the performance of the Dense-AutoGAN model was evaluated using the data set of malate dehydrogenase (MDH). Experimental results show that the accuracy of the model is improved after the addition of the encoder - decoder, and the stability of the model is significantly improved after the unidirectional superposition of the generator neural network. Dense-AutoGAN generates more accurate sequences with less loss, and generates new sequences with natural physical properties.

2. Materials and methods

2.1. Architecture of Dense-AutoGAN

The overall architecture of Dense-AutoGAN uses a generator network and a discriminator network of GAN as a framework. Feed the output of the Attention mechanism into the encoder - decoder, and the generator network is superimposed in the generator network, as shown in Figure 1. The generator network of Dense-AutoGAN uses a given protein sequence to generate new sequences, while the discriminator of Dense-AutoGAN is a classification network to distinguish generated sequences from true sequences. The two networks play a game learning. The discriminator promotes the generator network to produce fairly good output and generate generating sequences that approximate real sequences.

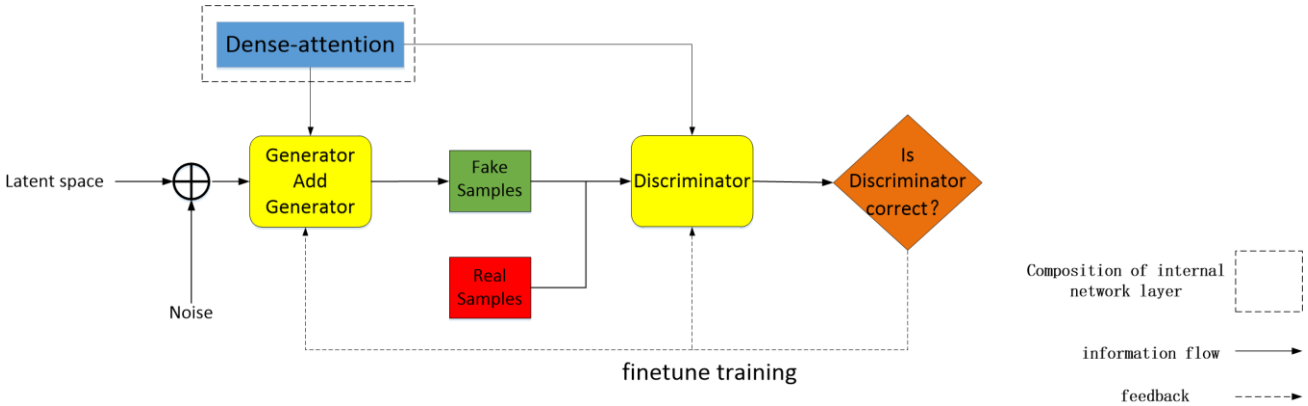


Figure 1. Overall Architecture of Dense-AutoGAN.

As shown in Figure 1, the solid arrows represent information flow. The virtual arrows represent feedback that allows the generator to fine-tune. The dotted boxes represent the composition of the internal network layer. Discriminators are used to distinguish between true and false sequences. The goal of the generator is to generate false sequences that are very close to deceive the discriminator. By selecting the elements in the potential space of training data to combine, and adding random noise, it is used as a false sequence. The original data and noise data are input into the generating network of GAN, and the data generated by the generating network and the original data are sent into the discriminator network together. The discriminator network judges the two types of data, so it is necessary to judge whether the discriminator network judges correctly. In the training process, if the discriminator network in GAN judges the generated data and real data correctly, it indicates that the data generated by the generated network is not close to the "real data". Therefore, it is necessary to adjust the parameters of the generated network to make the generated false data more realistic. The discriminator network judges the generated data as "real data" and outputs it as "real data".

In order to more effectively capture the characteristic information expressed by the multifunctional protein sequences, Dense-AutoGAN feeds the self-attention mechanism into the encoder-decoder frame structure. The transmission channel of the encoder and decoder is composed of the Dense network layer [44]. The number of denser neurons in the encoder is {256,128,64,32}, and the number of denser neurons in the decoder is {64,128,256}. The Dense layer mines the protein sequence information more thoroughly. The encoder-decoder receives the Attention mechanism output and learns about the potential relationships between protein sequences. The Attention mechanism and encoder-decoder network were fused into each network layer of the discriminator network and generator network. Each sequence vector was assigned different weights and mapped to feature space to obtain different feature information. The complex local features of the sequence are preserved completely, and the details of the sequence are easily extracted.

The Dense-AutoGAN discriminator network promotes the generator network to learn against it, so the generated protein sequence is closer to the original sequence. Discriminator and generator are expressed differently, and good classification effect is obtained by sigmoid activation function. It is helpful to improve the learning ability of generator to obtain the discriminator network with high accuracy by superimposing discriminator.

2.1.1. Generative adversarial network

The protein sequence vector is input into the generative network to capture the data rules of the protein sample sequence. The generated protein sequence is judged by the truth value after the maximum likelihood training of the input data, and it finally becomes the protein sequence with the specified distribution. Maximum likelihood function for the generator networkis shown in Eq (1):

$$\hat{q} = \arg\max_{q} P(x \mid q) = \arg\max_{q} \prod_{i=1}^{n} P(x_i \mid q). \tag{1}$$

Generator network and discriminator network are two sides of each other's game. Optimizing the generator network, the discriminator will confuse the generated protein sequence with the real one. By optimizing the discriminator network, the real sequence and generated protein sequence can be recognized to the greatest extent.

In adversarial learning and training, the objective function formula is expressed as Eq (2):

$$\min_{G}\max_{D}V(D,G)=E_{x\sim Pdata(x)}\left[\log D(x)\right]+E_{z\sim Pz(z)}\left[\log\left(1-D(G(z))\right)\right] \tag{2}$$

In Eq (2), G represents the generator; D represents the discriminator; V is a defined value function; E represents expectations. $x\sim Pdata(x)$ is the distribution of real protein sequence, x is the real protein sequence sample of $Pdata(x)$, $z\sim Pdata(z)$ is the noise sequence distribution of the generator networkand z is the noise data of $Pz(z)$. $G(z)$ represents the sequence generated by the generator network. $D(x)$ represents the probability that the discriminator network judges whether the real protein sequence is true. $D(G(z))$ is the probability that the discriminator network judges whether the generated protein sequence is true or false.

For the $V(D,G)$ function, the parameter set of G is regarded as a symbolic constant, and the parameter set D of the discriminator is regarded as a variable to find the maximum value of the function V, which is the algebraic expression of the Symbolic Constant containing G. At this time, the optimal parameter value of the generator can be obtained by finding the minimum value of the function.

2.1.2. Attention connection encoder-decoder

The so-called Attention mechanism is to calculate the weight of the hidden state of the encoder to the decoder at each time step by introducing a neural network, and finally calculate the output of the weighted encoder. Find associations between protein sequences based on existing data. Some important features of the sequence are highlighted based on weights to ensure that the model can capture important information about the protein. The Attention mechanism ignores the distance between input protein sequences. In the calculation process, any two protein sequences are directly linked through a calculation step, which can directly calculates the dependence between two sequences. The distance between long-distance dependent features is greatly shortened. The Attention mechanism enables the network to learn the internal structure of protein sequences effectively and capture sequence features effectively. The formula for the Attention mechanism is shown in Eq (3).

$$Attention(Q;K;V)=soft\max\left(\frac{QK^{T}}{\sqrt{d_k}}\right)V \tag{3}$$

$Q(Query)$, $K(Key)$ and $V(Value)$ matrices are all from the same input. By calculating the point multiplication between Q and K, they will be divided by a scale $\sqrt{d_k}$ to prevent the result from being too large, where d_k is the dimension of the $Query$ and Key vectors. The result is

normalized to the probability distribution by *Soft*max operation, and finally multiplied by V matrix to obtain the representation of weight summation.

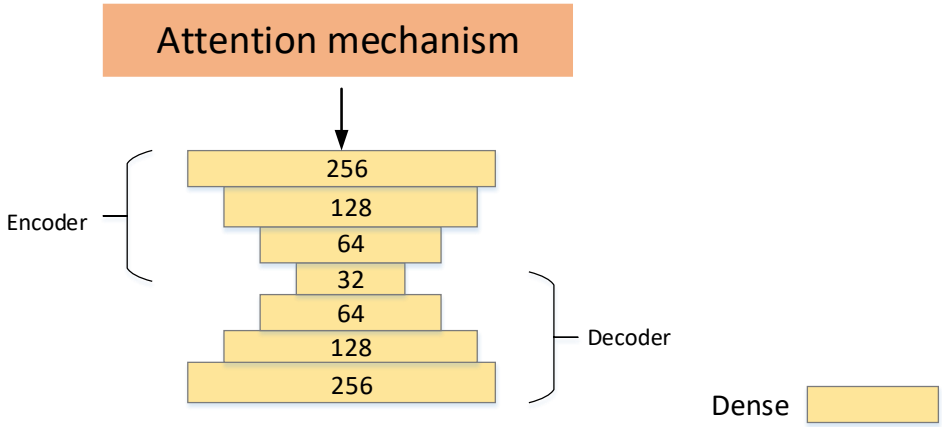
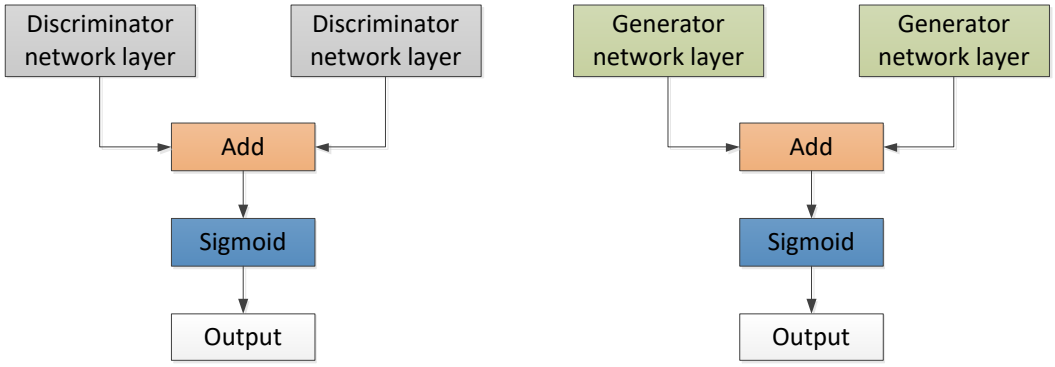


Figure 2. Connection structure of attention and density layers.

The better the discriminator network is trained, the worse the gradient disappearance is generated. Then, it becomes difficult to train the encoder-decoder. Therefore, the output of the Attention mechanism is encoded and decoded again to output an indefinite sequence. The encoder is used to process the protein sequence. It compresses the feature information of the sequence into a fixed length vector to analyze the sequence, and then generate the output sequence through the decoder. The Dense layer fills the encoder-decoder framework, and each node of the Dense layer is connected to these characteristic nodes to form the full connection layer. The feature information obtained from the output of the upper layer passes through the nonlinear changes in the dense layer, so that the associations between these features can be extracted and then mapped to the output space. In theory, it could be done in one dense layer. However, we do not know how many nodes are needed for a dense layer and how many times of training are needed. Therefore, adding dense layer can achieve faster convergence. Especially in the GAN network, we add a dense layer to improve the stability of the model.

Protein sequences are classified using fully connected layers. Each feature node of the Dense layer occupies a different proportion of weight, which is used to judge the classification of the input sequence. Finally, the classification prediction of the input will be jointly determined by all the features. In Dense layer, the relationship between these protein sequence features was extracted through nonlinear changes and mapped to the output space. The single-layer Dense network is not enough to determine the required training conditions, while the multi-layer Dense network can converge faster and solve the problem of non-convergence of GAN.

2.1.3. Stack generator computes convolutional neural network



(a) Internal structure of stack discriminator (b) Stack generator internal structure

Figure 3. Superposition structure diagram of generator and discriminator network.

In the process of protein sequence training, the GAN model will generate similar samples for different inputs during multiple trainings due to the complex molecular structure. As a result, the discriminator will not be able to correctly distinguish the model, and the training will be forced to stop. As shown in Figure 3(a) and 3(b), "Dual_gan" is the same computation processing for discriminator network and generator network at the same time. The GAN follows the principle of symmetry. The generator network and discriminator network are superimposed on each other in a symmetric form in the GAN model to expand the operation window of a sequence and attempt to alleviate the mode collapse problem of GAN. During the training of "Dual_gan," the loss value of the generator network continued to rise, and the phenomenon of mode collapse became more obvious. The complexity of a generator network and discriminator network are quite different. The discriminator network is in a dominant position, which leads to the decline of generator network performance and affects the diversity of protein sequence generation. It is necessary to continue to weaken the complexity of the discriminator network and improve the role of the generator network in GAN.

In symmetric GAN structures, the discriminator of GAN will be trained so well that it will affect the training of the generator. The generator gradient disappears, and the generator loss rises. Therefore, asymmetric structure is selected to reduce the network layer of the discriminator and improve the performance of the generator. As shown in Figure 3(b), only the one-way stack generator is used, and the network layers of the discriminator are not added. A layer of normalization is introduced into the generator to pull data distribution to the unsaturated region of the activation function. The layer of Normalization has weight scaling invariance and data scaling invariance to mitigate gradient explosion and accelerate convergence. We select the learning rate of the adjustment model and set the learning rate of the discriminator to 0.00005 and the generator to 0.0002. The generator network can obtain high accuracy, restrain molecule generation and improve the learning ability of the generator network. Without changing the speed, more information can be obtained, and confidence interval can be increased. The protein sequence information of both sides is fused to maximize the similarity of the generated sequence.

2.2. Dataset

Malate dehydrogenase (MDH) is a REDOX enzyme in lysosomes that catalyzes the reversible reaction of oxidative dehydrogenation of malic acid to oxaloacetate. There are two isozymes of MDH in human tissues, i.e., M-MDH in mitochondria and C-MDH in cytoplasm. M-mdh mainly catalyzes the dehydrogenation (oxidation) of malate in mitochondria, and is one of the important enzymes in the tricarboxylic acid cycle and plays an important role in the complete oxidation or mutual transformation of nutrients in the body. MDH is a white suspended liquid and can be stored at 2 ~ 8℃. The MDH sequence selected in this paper was downloaded from Uniprot on June 10, 2021. Sequences longer than 512 amino acids or containing non-canonical amino acids were filtered out. Finally, 16,898 sequence data sets were obtained, and the final data set was composed of 16,898 sequences. MMseq2 tool [45] was used to cluster them into 70% identical clusters during training to balance the data set. 20% of the clusters with less than 3 sequences were randomly selected for validation (192 sequences), and the remaining dataset was used for training (16,706 sequences) [46].

2.3. Parameter configuration

Dense-AutoGAN is uses Tensorflow (https://www.tensorflow.org). In the process of model training, the Adam optimization method is used, and the initial learning rate is 2E–4 in the generator network and 5E–5 in the discriminator network. In each epoch, due to GPU memory limitations, both networks used 16 batch sizes. We trained the discriminator network 99999 times and the generator network once at the same time. Dense-AutoGAN runs on 1 Nvidia 1080 Ti GPU and takes about 144 hours (20–30 periods) to converge. The performance of the model is evaluated by various training indexes in the training process. The Dense-AutoGAN model will generate 64 sequences, and the generative capacity of the model will be further determined by analyzing the training results.

2.4. Evaluation criteria

2.4.1. Loss function

As shown in Eq (4), when training the discriminator network, $D(x)$ in the first item means to distinguish the real samples. When the discrimination result of $D(x)$ is closer to 1, the value of the loss function is larger. In the second term, z is a random vector, and $G(z)$ represents the generated sample. For the generated samples, the discrimination result $D(g(z))$ of the discriminator network is closer to 0, and the final value is as large as possible.

$$\max_D V(D,G) = E_{x\sim Pdata(x)}\left[\log D(x)\right] + E_{z\sim Pz(z)}\left[\log\left(1-D(G(z))\right)\right] \tag{4}$$

As shown in Eq (5), when training the generator network, it is expected that the discriminator network will not recognize false samples as much as possible, that is, $D(g(z))$ is as close to 1 as

possible. Then, $\log\ (1-D\ (g\ (z)))$ is as large as possible, and the final value is as small as possible.

$$\log\ (1-D\ (g\ (z))) \tag{5}$$

2.4.2. Gradient penalty

The gradient penalty is shown in Eq (6). As a regularizer, this function can scale the loss. The gradient penalty is used to constrain the loss function to meet the 1-Lipschitz condition the value of the loss function is sandwiched between ($y=x$ and $y=-x$), so that the gradient of the model doesn't explode or disappear.

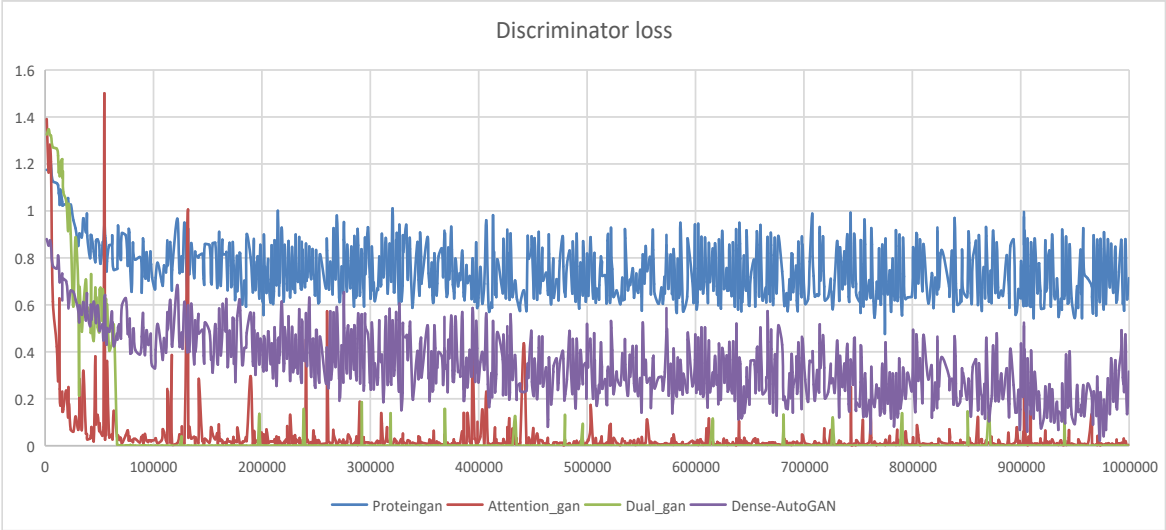
$$gradient\ penalty=\lambda E\left[\left(\left\|\nabla_{\hat{x}}D\left(\hat{x}\right)\right\|_2-1\right)^2\right] \tag{6}$$

3. Results and discussion

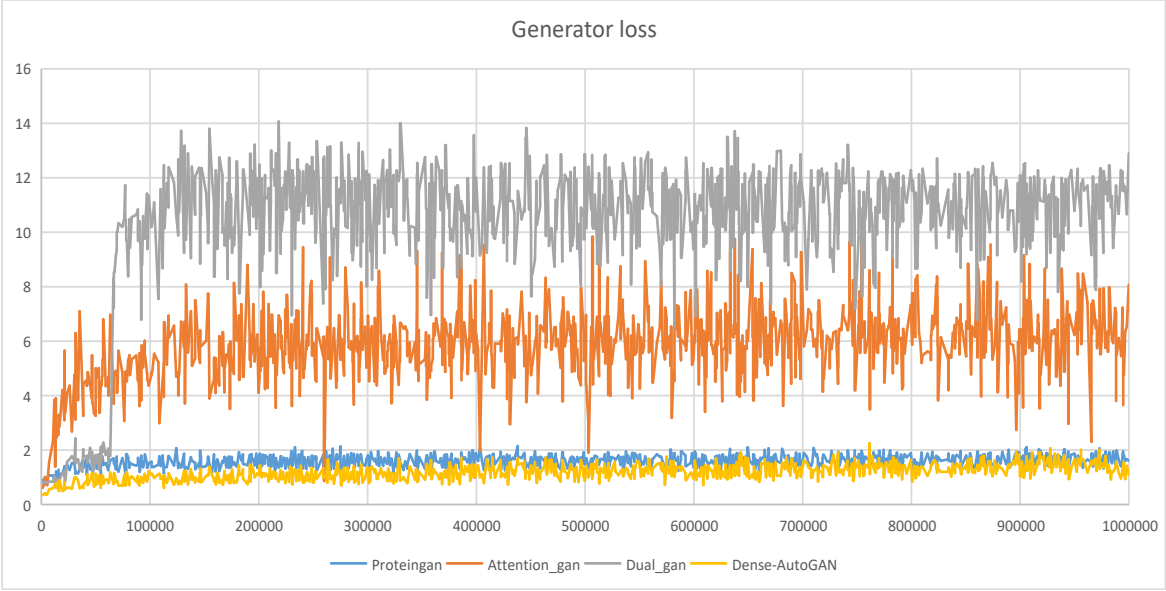
During the GAN training process, the training task of the generator progressed slowly. According to the results of training, we check and adjust the structure and parameters of the model to improve the performance of the model. ProteinGAN is the prototype network model, which provides comparison and reference. Attention_GAN sends the Attention mechanism output embedded in the GAN network into the Dense fully connected layer to improve the accuracy of the generated molecules. Dual_GAN performs double stack operation on a generator and discriminator network to improve the stability and accuracy of the internal neural network. Dense_AutoGAN is combined with Attention_GAN to retain the one-side stacking operation of Dual_GAN and improve the stability.

3.1. Loss curve analysis of model training

When the support for the model distribution does not intersect with the support for the target distribution (real image), there is a discriminator that can distinguish the model distribution from the target distribution well. In Figure 4, it can be seen that the discriminator derivative of the model Attention_GAN and Dual_GAN after training gradually converges to 0, so that the training of the generator gradually stops. If Attention_gan and Dual_gan continue to be trained, the discriminator loss of these two models will eventually become zero, and the Generator loss will continue to fluctuate. This state causes the model to generate gradient exploding, making the training of GAN very difficult. In addition, the increase of Generator loss of Dual_GAN is larger than Attention_GAN, so the network layer of the discriminator network needs to be appropriately reduced. The discriminator loss and generator loss of Dense-AutoGAN are low, and the loss curve is lower than that of the ProteinGAN network. Therefore, appropriate adjustment of the learning rate of Dense-AutoGAN will affect the loss of the model.



(a) Discriminator loss



(b) Generator loss

Figure 4. Model training loss curve: (a) discriminator training loss function curve; (b) loss function training curve of generator.

3.2. Results and analysis after introducing gradient penalty

Table 1. Mean of the gradient penalty obtained by the four models.

Model	ProteinGAN	Attention_GAN	Dual_GAN	Dense-AutoGAN
gradient penalty (Average)	0.0385	0.0005	0.0012	0.0276

In the process of protein training, gradient punishment is a method to make the loss function meet the 1-Lipschitz condition and reduce the probability of pattern collapse. Gradient Penalty can significantly improve the training speed and solve the problem of slow GAN convergence. Table 1 shows the mean value of gradient punishment of the four models. The lowest value of Dual_GAN is 0.0005, which is close to 0, and Attention_GAN is also close to 0. The poor convergence performance of Dual_GAN and Attention_GAN models has affected the training of protein molecules in the model. The gradient penalty of Dense-AutoGAN was 0.0276, which showed the best convergence performance.

3.3. Comparison and analysis of discriminator scores curve during training

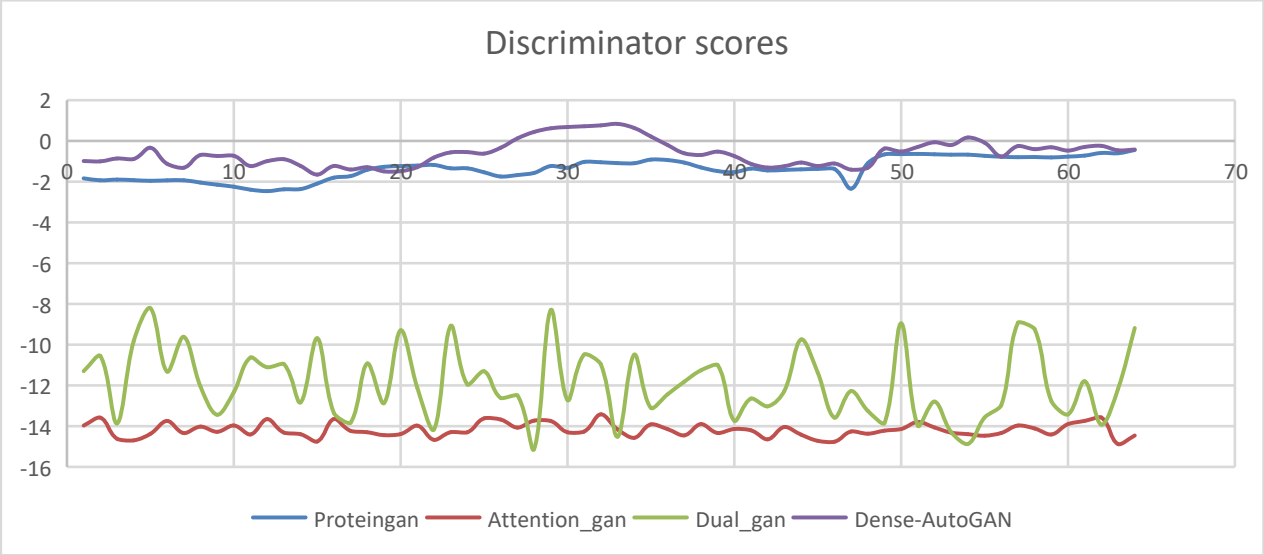


Figure 5. Discriminator score curves of four models during training.

In Figure 5, the discriminator scores curve shows that Attention_GAN has a low score, but has strong stability and little fluctuation, which is conducive to model convergence. While Dense-AutoGAN has a high score for a long time, the Layer Normalization neuron input has the same mean and variance. Different input samples have different mean and variance. The data are normalized one by one and placed in the generator. It helped the generator network to improve the expression ability of the model, stabilized the Dense-AutoGAN model and made the generated protein characteristics more in line with the requirements.

3.4. Comparison and analysis on variation of sequences results

Table 2. Mean true/false values of sequence variation in the four models.

Model		**ProteinGAN**	**Attention_GAN**	**Dual_GAN**	**Dense-AutoGAN**
Variation of sequences (Average)	Real	0.1066	0.1066	0.1101	0.0966
	Fake	0.1006	0.0112	0.0990	0.0917

In Table 2, only the Real samples of Attention_GAN and ProteinGAN are consistent in their Variation of sequences with the lowest Variation ratio, while the Real samples of Dual_GAN have the highest Variation ratio. The proportion of Dense-AutoGAN Fake sample Variation was the highest, but the difference between Dense-AutoGAN and ProteinGAN was small. The proportion of Attention_GAN Fake sample Variation was the lowest. Increasing the complexity of Dual_GAN model will increase the proportion of Real sample Variation and reduce the proportion of Fake sample Variation. Dense-AutoGAN reduces the degree of sequence variation by increasing the complexity of generators.

3.5. Comparison and analysis of protein sequences generated by four models

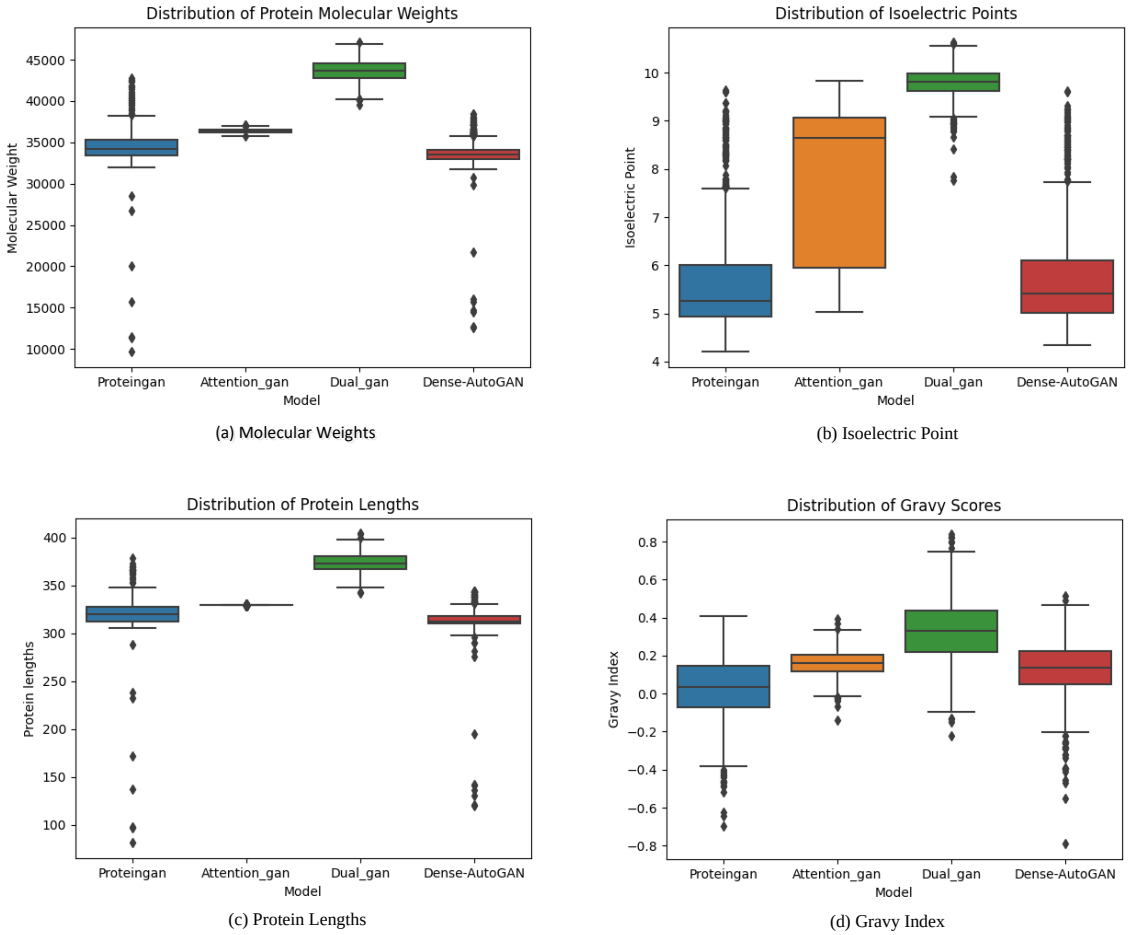


Figure 6. The comparison of Protein sequences of four models in Molecular Weights, Isoelectric points, Protein lengths and Gravy index.

Figure 6 shows the performance of the 4 models with Molecular Weights, Isoelectric Point, Protein lengths and Gravy index. In these 4 models, Dual_GAN obtained the highest value. Due to the well-trained discriminator, the sequence generation of the model was affected, so the parameters of the generated protein were too high.

According to the Box-plot in Figure 6(a), the Molecular Weights of Dense-AutoGAN are concentrated between 3000 and 3500, showing a normal distribution. According to the Box-plot in

Figure 6(b), the Isoelectric Point of Dense-AutoGAN was between Figures 5 and 6. The protein generated by Dense-AutoGAN was negatively charged and showed a left skewed distribution. According to the Box-plot in Figure 6(c), the Protein lengths of Dense-AutoGAN were lower than other three models. The length of sequences generated by Dense-AutoGAN is relatively concentrated. The most computationally complex Dual_gan had the longest protein sequence. Attention_gan generates single-length protein sequences. Moreover, Protein lengths are improved by stacking discriminator networks and generator networks in GAN. Protein lengths can be effectively reduced by the one-way stack generator network in GAN. According to the Box-plot in Figure 6(d), Dual_GAN had the highest protein marinade index. Dense-AutoGAN has a Gravy index between 0 and 0.2. They are hydrophobic and amphoteric amino acids, which can be used to predict transmembrane helices. The protein sequences generated by Dense-AutoGAN are mostly hydrophobic proteins with low molecular weight between 3000–3500 and Isoelectric point between Figures 5 and 6.

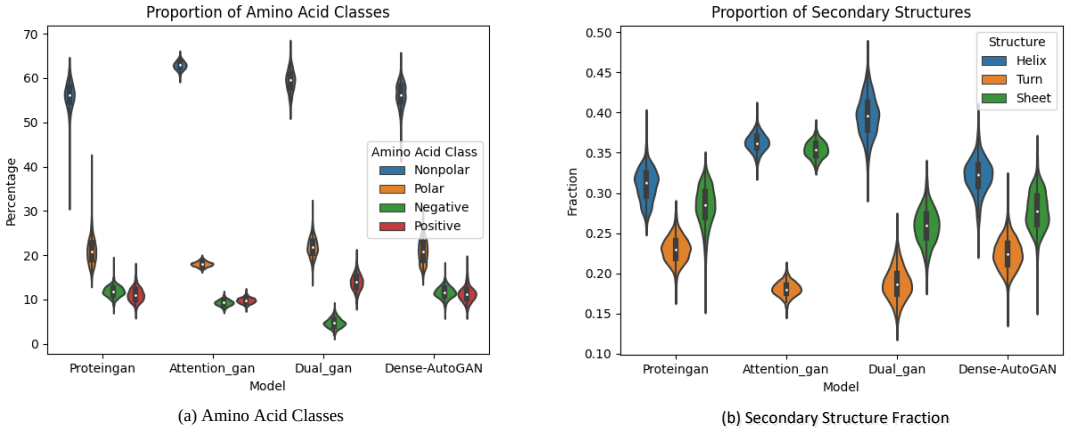


Figure 7. Compare Amino Acid Classes and Secondary structure fraction.

Figure 7 shows the representation of Amino Acid Classes and Secondary structure fraction of the four models. Figure 7(a), Amino Acid Classes, consist of nonpolar, polar, negative and positive.

In Figure 7(a), the upper limit of negative charge of Dual_GAN is close to the lower limit of its own positive charge. It has the highest proportion of non-polar amino acids. Therefore, Dual_GAN has high hydrophobicity. Attention_GAN has slightly more positive charges than negative charges. Attention_GAN has the lowest proportion of polar amino acids and the highest proportion of polar amino acids lowest proportion of polar amino acids and the highest proportion of polar amino acids. Therefore, Attention_GAN shows high hydrophilicity. The ratio of non-polar amino acids in Dense-AutoGAN was 50–70 percent, and the mean value was the lowest. The solubility of Dense-AutoGAN in water was lower than that of polar amino acids, and its hydrophobicity increased with the increase of aliphatic side chain length. The proportion of polar amino acids in the sequences generated by Dense-AutoGAN is 10% to 30%. Therefore, the sequences generated by this model are hydrophilic and can combine with suitable molecules. Dense-AutoGAN has a similar proportion of amino groups with positive charge and carboxyl groups with the ability to dissociate into negative charge, and it can undergo normal amphotericity ionization.

As shown in Figure 7(b), Secondary structure fraction includes alpha-helix, beta-sheet and reverse

turn. Among them, the proportions of the three attributes of the secondary structure of Dense-AutoGAN were balanced, and Dense-AutoGAN was more conducive to the generation of alpha-helix than proteinGAN. There is a large gap between Attention_GAN and Dual_GAN. Dual_GAN is more conducive to the formation of alpha-helix, while Attention_GAN is not conducive to the formation of reverse turn.

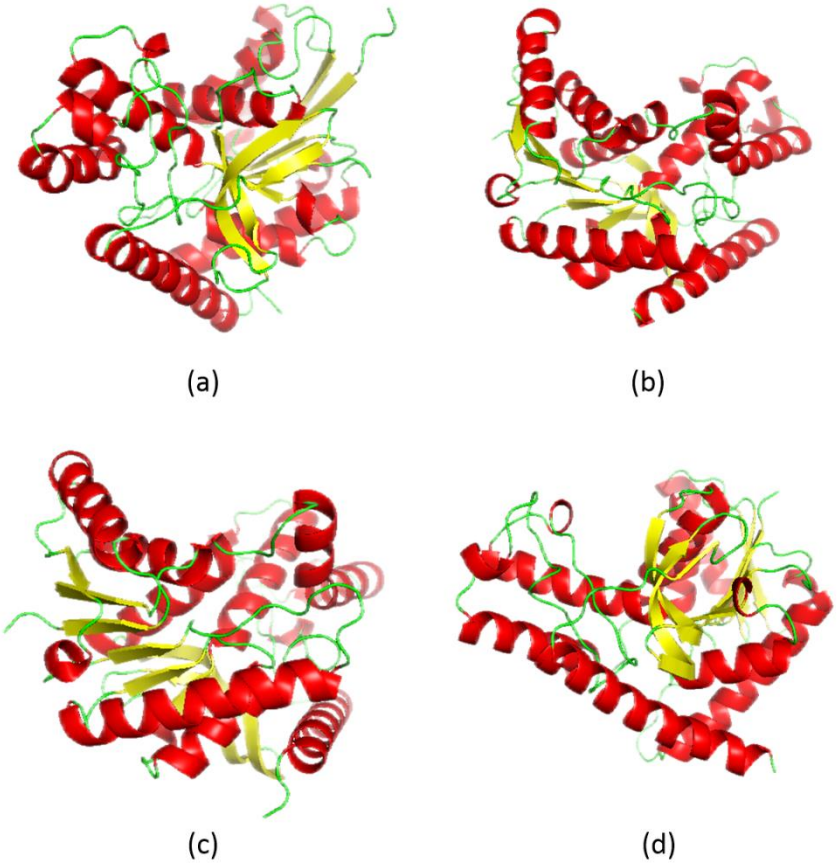


Figure 8. Dense-AutoGAN generated 3D structure of 4 random sequences.

In Figure 8(a)–(d) are the 3D structures of four randomly generated sequences predicted by the AlphaFold2 [47] method. In AlphaFold2, the index to evaluate the reliability of monomer structure prediction is pLDDT, which ranges from 0 to 100. pLDDT corresponds to the model's predicted per-residue scores on the metric $lDDT-C\alpha$.The larger the value, the more reliable the predicted structure is. When the value of pLDDT is above 90, the reliability is considered very high. When the value of pLDDT is below 90 and above 70, it is considered confident. When the value of pLDDT is lower than 70, the reliability is considered to be low. When the value of pLDDT is below 50, it is basically considered to be very low credibility, and the generated sequence is disordered sequence prediction.

Table 3. pLDDT values of four random sequences generated by Dense-AutoGAN.

pLDDT	(a)	(b)	(c)	(d)
Value	97.02	94.73	95.84	96.36

In Table 3, it can be seen that the reliability of four sequences generated by random extraction of Dense-AutoGAN is greater than 90, which proves that the reliability of sequences generated by this model is very high.

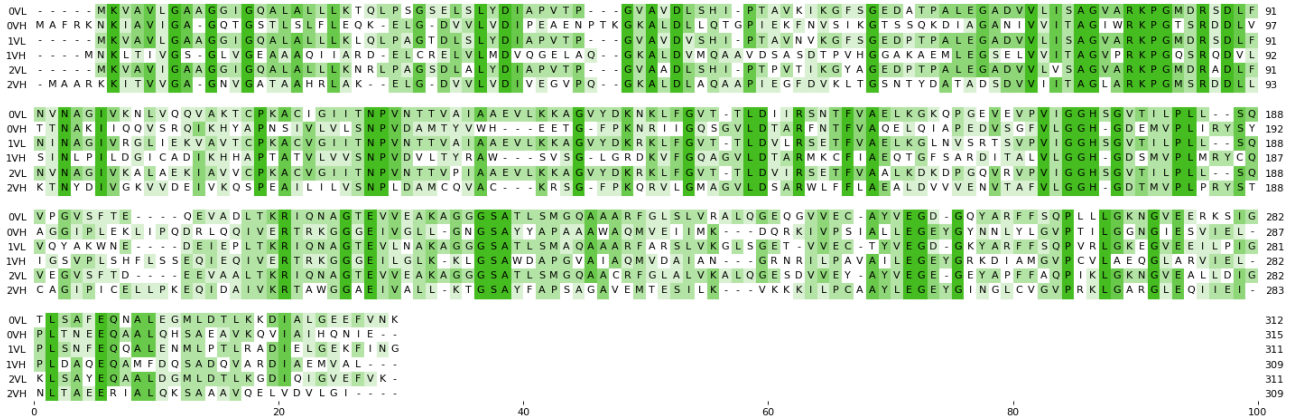


Figure 9. Alignment results between the training set sequence and Dense-AutoGAN generated sequence.

In Figure 9, multiple sequence alignment (MSA) is carried out between training set (VL) and Dense-AutoGAN generated sequence (VH), and various mutation events can be depicted by visual narration. The first three groups of structures of the sequences generated by the two models represented in Figure 9 can be seen from the sequence alignment results that the two groups of sequences are partially matched to generate protein sequences with similar properties, so the generated proteins often have similar functions.

3.6. Improved models of multi-generator and multi-discriminator

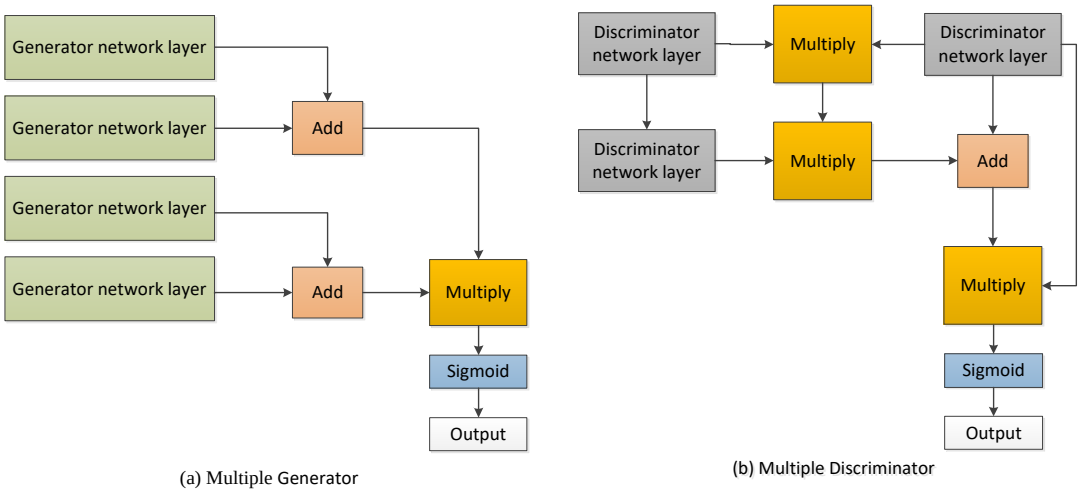


Figure 10. Network structure diagram of multi-generator and multi-discriminator.

In Figure 10, the multiple generator networks and discriminator networks are used to improve the stability of generated molecules by increasing the spatial complexity of computation. In the process of processing generator network, protein sequence information was fused by adding two generator networks and multiplying them. The two discriminator networks in GAN are multiplied to fuse feature information. After information fusion, the output vector is directly multiplied with the vector input for the first time, and the feature information is continued to be fused. Finally, linearly transform the vector output by the upper stage. Through the above calculation methods, the accuracy of the discriminator network and the learning ability of the generator network are improved. According to Figure 11, product fusion information is added to GAN network to improve the complexity. It is found that the loss curve of discriminator gradually converges to 0, and the loss curve of generator rises to about 16. The model has a serious mode collapse. The discriminator score curve is shown in Figure 12, with an average discriminator score of –14.188. The discriminator score fluctuates greatly, and GAN is not suitable for more complex operations.

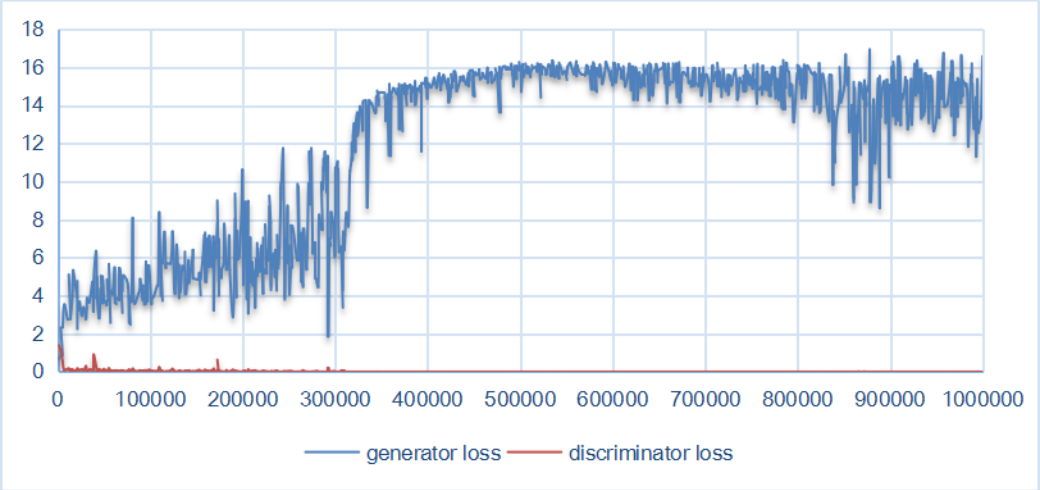


Figure 11. Loss function curve of multiple generators and discriminators.

Figure 12. Discriminator Score curve variation.

4. Conclusions

In de novo design of proteins, it is very important to generate efficient protein sequences. Protein generation has made great progress with the support of deep learning, which needs to further optimize the generation model to obtain better generation results. In this study, we propose a new GAN-based architecture called Dense-AutoGAN for protein sequence generation. Each layer of Dense-AutoGAN incorporates the attention mechanism, and the network layer combined with the encoder and decoder of Dense-AutoGAN helps the model accurately amplify local feature information. By changing the loss function of the GAN network, the imbalance problem can be solved, and the antagonistic learning ability of the model can be enhanced to the greatest extent. In addition, during adversarial learning, Dense-AutoGAN uses a superimposed generator network for computation to improve model generation and obtain more information. This method not only avoids the mode collapse of GAN network, but also restricts the unbalanced state to some extent.

In conclusion, we propose a GAN-based deep learning architecture for sequence generation, which can effectively improve the generation capability of models. We will complete this research work through various verification experiments in the future. We'd like to thank the company of Primary Biotech (www.pumeirui.com) for the support of protein structure modeling.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (81603152) and the fund of Changzhou Sci. and Tech. Program (CE20205033).

Conflict of interest

The authors declare there is no conflict of interest.

References

1. E. C. Alley, G. Khimulya, S. Biswas, M. AlQuraishi, G. M. Church, Unified rational protein engineering with sequence-based deep representation learning, *Nat. Methods*, **16** (2019), 1315–1322. https://doi.org/10.1038/s41592-019-0598-1
2. K. K. Yang, Z. Wu, F. H. Arnold, Machine-learning-guided directed evolution for protein engineering, *Nat. Methods*, **16** (2019), 687–694. https://doi.org/10.1038/s41592-019-0496-6
3. M. Rouhani, F. Khodabakhsh, D. Norouzian, R. A. Cohan, V. Valizadeh, Molecular dynamics simulation for rational protein engineering: Present and future prospectus, *J. Mol. Graph. Model.*, **84** (2018), 43–53. https://doi.org/10.1016/j.jmgm.2018.06.009
4. B. A. Meinen, C. D. Bahl, Breakthroughs in computational design methods open up new frontiers for de novo protein engineering, *Protein Eng. Des. Sel.*, **34** (2021). https://doi.org/10.1093/protein/gzab007
5. I. V. Korendovych, W. F. DeGrado, De novo protein design, a retrospective, *Q. Rev. Biophys.*, **53** (2020). https://doi.org/10.1017/S0033583519000131
6. E. Marcos, D. A. Silva, Essentials of de novo protein design: Methods and applications, *WIREs Comput. Mol. Sci.*, **8** (2018), e1374. https://doi.org/10.1002/wcms.1374

7. Y. Hsia, R. Mout, W. Sheffler, N. I. Edman, I. Vulovic, Y. Park, et al., Design of multi-scale protein complexes by hierarchical building block fusion, *Nat. Commun.*, **12** (2021), 1–10. https://doi.org/10.1038/s41467-021-22276-z
8. J. O'Connell, Z. Li, J. Hanson, R. Heffernan, J. Lyons, K. Paliwal, et al., SPIN2: Predicting sequence profiles from protein structures using deep neural networks, *Proteins: Struct. Function Bioinform.*, **86** (2018), 629–633. https://doi.org/10.1002/prot.25489
9. Q. Zou, G. Lin, X. Jiang, X. Liu, X. Zeng, Sequence clustering in bioinformatics: An empirical study, *Brief. Bioinform.*, **21** (2020), 1–10. https://doi.org/10.1093/bib/bby090
10. Y. Li, C. Huang, L. Ding, Z. Li, Y. Pan, X. Gao, Deep learning in bioinformatics: Introduction, application, and perspective in the big data era, *Methods*, **166** (2019), 4–21. https://doi.org/10.1016/j.ymeth.2019.04.008
11. F. Gabler, S. Nam, S. Till, M. Mirdita, M. Steinegger, J. Söding, et al., Protein sequence analysis using the MPI bioinformatics toolkit, *Curr. Protoc. Bioinform.*, **72** (2020), e108. https://doi.org/10.1002/cpbi.108
12. Kaitao Lai, Natalie Twine, Aidan O'Brien, Yi Guo, Denis Bauer, Artificial intelligence and machine learning in bioinformatics, *Encycl. Bioinform. Comput. Biol.*, **1** (2019), 272–286. https://doi.org/10.1016/b978-0-12-809633-8.20325-7
13. Z. Qin, L. Wu, H. Sun, S. Huo, T. Ma, E. Lim, et al., Artificial intelligence method to design and fold alpha-helical structural proteins from the primary amino acid sequence, *Extreme Mech. Lett.*, **36** (2020), 100652. https://doi.org/10.1016/j.eml.2020.100652
14. T. Kirioka, P. Aumpuchin, T. Kikuchi, Detection of folding sites of β-trefoil fold proteins based on amino acid sequence analyses and structure-based sequence alignment, *J. Proteomics Bioinform.*, **10** (2017), 222–235. https://doi.org/10.4172/jpb.1000446
15. B. Liu, BioSeq-Analysis: A platform for DNA, RNA and protein sequence analysis based on machine learning approaches, *Brief. bioinform.*, **20** (2019), 1280–1294. https://doi.org/10.1093/bib/bbx165
16. S. Makrodimitris, R. C. van Ham, M. J. Reinders, Improving protein function prediction using protein sequence and GO-term similarities, *Bioinformatics*, **35** (2019), 1116–1124. https://doi.org/10.1093/bioinformatics/bty751
17. Z. J. Lee, S. Su, C. Chuang, K. Liu, Genetic algorithm with ant colony optimization (GA-ACO) for multiple sequence alignment, *Appl. Soft Comput.*, **8** (2008), 55–78. https://doi.org/10.1016/j.asoc.2006.10.012
18. M. Dorigo, M. Birattari, T. Stutzle, Ant colony optimization, *IEEE Comput. Intell. Mag*., **1** (2006), 28–39. https://doi.org/10.1109/MCI.2006.329691
19. S. Katoch, S. S. Chauhan, V. Kumar, A review on genetic algorithm: Past, present, and future, *Multimedia Tools Appl.*, **80** (2021), 8091–8126. https://doi.org/10.1007/s11042-020-10139-6
20. B. Bošković, J. Brest, Protein folding optimization using differential evolution extended with local search and component reinitialization, *Inform. Sci.*, **454** (2018), 178–199. https://doi.org/10.1016/j.ins.2018.04.072
21. M. Issa, A. E. Hassanien, D. Oliva, A. Helmi, I. Ziedana, A. Alzohairy, ASCA-PSO: Adaptive sine cosine optimization algorithm integrated with particle swarm for pairwise local sequence alignment, *Expert Syst. Appl.*, **99** (2018), 56–70. https://doi.org/10.1016/j.eswa.2018.01.019

22. H. Nenavath, R. K. Jatoth, Hybridizing sine cosine algorithm with differential evolution for global optimization and object tracking, *Appl. Soft Comput.*, **62** (2018), 1019–1043. https://doi.org/10.1016/j.asoc.2017.09.039
23. P. Dutta, S. Saha, S. Naskar, A multi-objective based PSO approach for inferring pathway activity utilizing protein interactions, *Multimedia Tools Appl.*, **80** (2021), 30283–30303. https://doi.org/10.1007/s11042-020-09269-8
24. Q. Zhan, N. Wang, S. Jin, R. Tan, Q. Jiang, Y. Wang, Probpfp: A multiple sequence alignment algorithm combining partition function and hidden Markov model with particle swarm optimization, in *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, (2018), 1290–1295. https://doi.org/10.1109/BIBM.2018.8621220
25. H. Rademacher, On the partition function p (n), *Proc. London Math. Soc.*, **2** (1938), 241–254. https://doi.org/10.1112/plms/s2-43.4.241
26. N. Plattner, S. Doerr, G. de Fabritiis, F. Noé, Complete protein–protein association kinetics in atomic detail revealed by molecular dynamics simulations and Markov modelling, *Nat. Chem.*, **9** (2017), 1005–1011. https://doi.org/10.1038/nchem.2785
27. C. Yang, Y. Lin, L. Chuang, H. Chang, A particle swarm optimization-based approach with local search for predicting protein folding, *J. Comput. Biol.*, **24** (2017), 981–994. https://doi.org/10.1089/cmb.2016.0104
28. M. F. M. Silva, L. F. Leijoto, C. N. Nobre, Algorithms analysis in adjusting the SVM parameters: An approach in the prediction of protein function, *Appl. Artif. Intell.*, **31** (2017), 316–331. https://doi.org/10.1080/08839514.2017.1317207
29. Y. Ge, S. Zhao, X. Zhao, A step-by-step classification algorithm of protein secondary structures based on double-layer SVM model, *Genomics*, **112** (2020), 1941–1946. https://doi.org/10.1016/j.ygeno.2019.11.006
30. K. C. Chou, L. Carlacci, Simulated annealing approach to the study of protein structures, *Protein Eng. Des. Sel.*, **4** (1991), 661–667. https://doi.org/10.1093/protein/4.6.661
31. C. Yanover, M. Fromer, J.M. Shifman, Dead-end elimination for multistate protein design, *J. Comput. Chem.*, **28** (2007), 2122–2129. https://doi.org/10.1002/jcc.20661
32. A. W. Senior, R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, et al., Improved protein structure prediction using potentials from deep learning, *Nature*, **577** (2020), 706–710. https://doi.org/10.1038/s41586-019-1923-7
33. M. Gao, J. Skolnick, A novel sequence alignment algorithm based on deep learning of the protein folding code, *Bioinformatics*, **37** (2021), 490–496. https://doi.org/10.1093/bioinformatics/btaa810
34. M. Spencer, J. Eickholt, J. Cheng, A deep learning network approach to ab initio protein secondary structure prediction, *IEEE/ACM Trans. Comput. Biol. Bioinform.*, **12** (2015), 103–112. https://doi.org/10.1109/TCBB.2014.2343960
35. P. Kim, Convolutional neural network, in *MATLAB Deep Learning*, Springer, (2017), 121–147. https://doi.org/10.1007/978-1-4842-2845-6_6
36. J. Hou, B. Adhikari, J. Cheng, DeepSF: Deep convolutional neural network for mapping protein sequences to folds, *Bioinformatics*, **34** (2018), 1295–1303. https://doi.org/10.1093/bioinformatics/btx780
37. S. Wang, J. Peng, J. Ma, J. Xu, Protein secondary structure prediction using deep convolutional neural fields, *Sci. Rep.*, **6** (2016), 1–11. https://doi.org/10.1038/srep18962

38. S. Wang, S. Weng, J. Ma, Q. Tang, DeepCNF-D: Predicting protein order/disorder regions by weighted deep convolutional neural fields, *Int. J. Mol. Sci.*, **16** (2015), 17315–17330. https://doi.org/10.3390/ijms160817315
39. N. Killoran, L. J. Lee, A. Delong, D. Duvenaud, B. J. Frey, Generating and designing DNA with deep generative models, preprint, arXiv:1712.06148v1. https://doi.org/10.48550/arXiv.1712.06148
40. E. Lin, C. H. Lin, H. Y. Lane, Relevant applications of generative adversarial networks in drug design and discovery: Molecular de novo design, dimensionality reduction, and de novo peptide and protein design, *Molecules*, **25** (2020), 3250. https://doi.org/10.3390/molecules25143250
41. C. Wan, D. T. Jones, Protein function prediction is improved by creating synthetic feature samples with generative adversarial networks, *Nat. Mach. Intell.*, **2** (2020), 540–550. https://doi.org/10.1038/s42256-020-0222-1
42. A. Gupta, J. Zou, Feedback GAN for DNA optimizes protein functions, *Nat. Mach. Intell.*, **1** (2019), 105–111. https://doi.org/10.1038/s42256-019-0017-4
43. M. R. Uddin, S. Mahbub, M. S. Rahman, M. S. Bayzid, SAINT: Self-attention augmented inception-inside-inception network improves protein secondary structure prediction, *Bioinformatics*, **36** (2020), 4599–4608.
44. G. Huang, Z. Liu, L. van der Maatene, K. Q. Weinberger, Densely connected convolutional networks. in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2017). https://doi.org/10.1109/CVPR.2017.243
45. M. Steinegger, J. Söding, MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets, *Nat. Biotechnol.*, **35** (2017), 1026–1028. https://doi.org/10.1038/nbt.3988
46. D. Repecka, V. Jauniskis, L. Karpus, E. Rembeza, I. Rokaitis, J. Zrimec, et al., Expanding functional protein sequence spaces using generative adversarial networks, *Nat. Mach. Intell.*, **3** (2021), 324–333. https://doi.org/10.1038/s42256-021-00310-5
47. J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, et al., Highly accurate protein structure prediction with AlphaFold, *Nature*, 596 (2021), 583–589.

AIMS Press